

Analiza wydajności złączeń i zagnieżdżeń dla schematów znormalizowanych i zdenormalizowanych, w różnych bazach danych

1. Wstęp

Bazy danych stanowią bardzo ważny element funkcjonowania dzisiejszego świata. Przechowanie informacji ma zastosowanie praktyczne w każdej dziedzinie współczesnego życia, od rejestracji w przychodni, zakupów w sklepie internetowym czy też składowaniu informacji o uprawnieniach do kierowania danymi pojazdami przez państwa. Poradzenie sobie z tak dużą ilością informacji wymaga sprzętu, technologii oraz wiedzy w jaki sposób sprawić, aby dane były bezpieczne, trwałe oraz dostęp do nich był szybki.

2. Określenie problemu

W przypadku przechowywania bardzo dużej ilości danych dotyczących danego zagadnienia należy zaprojektować ich składowanie w sposób optymalny. Jednym z najważniejszych czynników branych pod uwagę podczas projektowania gotowych systemów jest ich wydajność, czyli czas jaki należy odczekać od wysłania zapytania do bazy danych do momentu otrzymania odpowiedzi z rządymi informacjami. Schemat gwiazdy oraz schemat płatka śniegu to dwa najpopularniejsze koncepty składowania danych w hurtowniach danych, poniżej krótka charakterystyka każdej z nich:

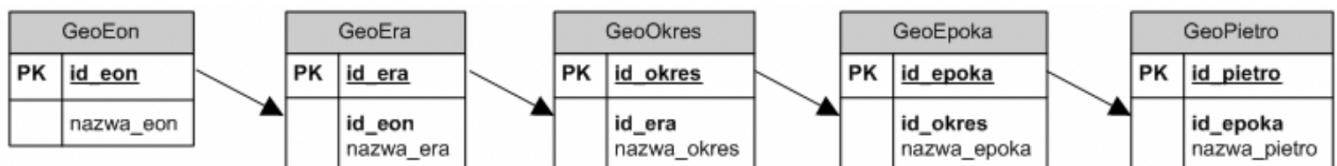
a). Schemat gwiazdy - najprostszy model projektu bazy danych, zakłada istnienie centralnej tabeli w postaci zdenormalizowanej, czyli tabeli faktów oraz połączonych z nią tabel wymiarów, będących w 2 postaci normalnej. Taka koncepcja jest gotowa na zwracanie prostych zapytań w szybkim tempie. Klucz główny w tabeli faktów najczęściej składa się z wszystkich jej kolumn z pominięciem wartości numerycznych, reprezentujących wymiary. Większość danych zawartych w tabeli faktów jest równocześnie przechowywana w tabeli wymiarów, celem szybkiego celu prostego i szybkiego przeglądania kategorii.

b). Schemat płatka śniegu - bardziej złożony i zaawansowany niż schemat gwiazdy. Tabele wymiarów również są znormalizowane. Jest stosowana gdy tabela wymiarów osiąga zbyt duże wymiary aby w elastyczny oraz optymalny sposób przechowywać w niej dane. Implementuje założenia schematów relacyjnych baz danych.

3. Prezentacja danych

Operacje testowe zostaną przeprowadzone na tabelach zawierających dane dotyczące podziału czasu w geologii Ziemi - tabeli stratygraficznej, oraz dwóch tabelach pomocniczych Dziesięć oraz Milion, zawierających odpowiednio dziesięć oraz milion rekordów.

Schemat tabeli stratygraficznej w postaci znormalizowanej, wykorzystywanej w schemacie płatka śniegu:



Rys.1. Schemat tabeli stratygraficznej w postaci znormalizowanej

Wykorzystując komendę:

```
CREATE TABLE GeoTabela AS (SELECT * FROM GeoPietro NATURAL JOIN GeoEpoka
NATURAL JOIN GeoOkres NATURAL JOIN GeoEra NATURAL JOIN GeoEon)
```

Tabela zostaje doprowadzona do postaci zdenormalizowanej i wykorzystana w schemacie gwiazdy, schemat poniżej:

GeoTabela	
PK	<u>id_pietro</u>
	nazwa_pietro
	<u>id_epoka</u>
	nazwa_epoka
	<u>id_okres</u>
	nazwa_okres
	<u>id_era</u>
	nazwa_era
	<u>id_eon</u>
	nazwa_eon

Rys.2. Schemat tabeli w postaci zdenormalizowanej

Zawartość tabeli stratygraficznej:

ERA	OKRES	EPOKA	WIEK (PIĘTRO)
HOLOCE			
	Q CZWARTORZĘD	NEO PL MEZO PL EO PL	WURM RISS-WURM RISS MINDEL-RISS MINDEL GUNZ-MINDEL GUNZ DONAU-GUNZ DONAU BIBER-DONAU BIBER
K O Z O N E K	TR TRZECIORZĘD	NEOGEN NG PALEOGEN PG	PLIOCEN MIOCEN OLIGOCEN EOCEN PALEOCEN
			ROMAN DAK PONT PANON SARMAT BADEN KARPAT OTNANG EGENBURG EGGER SZAT RUPEL PRIABON BARTON LUTET YPREZ TANET DAN

K	KREDA CR	GÓRNA	MASTRYCHT KAMPAN SANTON KONIAK TURON CENOMAN	CRM CRCP CRST CRCN CRT CRC
O		DOLNA	ALB APT BARREM HOTERYW WALANZYN BERIAS	CRAL CRAP CRBA CRH CRV CRB
Z	JURA J	GÓRNA (MALM)	TYTON KIMERYD OKSFORD	JT JKM JO
O		ŚRODKOWA (DOGGER)	KELOWEJ BATON BAJOS AALEN	JCL JBT JB JA
Z		DOLNA (LIAS)	TOARK PLIENSBACH SYNEMUR HETANG	JTO JPL JS JH
E	TRIAS T	GÓRNY	RETYK NORYK KARNIK	T3
M		ŚRODKOWY	LADYN ANIZYK	T2
		DOLNY	SCYTYK	T1
K	PERM P	GÓRNY	TATAR KAZAŃ KUNGUR	P3
I		DOLNY	ARTYNSK SAKMAR ASELSK	P1
O	KARBON C	GÓRNY	STEFAN WESTFAL NAMUR	CS CW CN
N		DOLNY	WIZEN TURNÉJ	CWI CT
E	DEWON D	GÓRNY	FAMEN FRAN	DFA DFR
		ŚRODKOWY	ZYWET EIFEL	DGT DE
		DOLNY	EMS ZIGEN ŽEDYN	DEM DZ DGD

Schemat tabeli Dziesięć Oraz milion:

Dziesięć	
Cyfra	Integer M
Bit	Integer M

Milion	
liczba	Integer M
cyfra	Integer M
bit	Integer M

Rys. 3. Schemat tabeli Dziesięć

Rys. 4. Schemat tabeli

Utworzenie tabeli Milion za pomocą odpowiedniego auto złączenia tabeli Dziesięć wypełnionej liczbami 0-9.

```
--tabela Milion
CREATE TABLE Milion(
    liczba INT,
    cyfra INT,
    bit INT
)

--dodane wartości
INSERT INTO Milion SELECT a1.cyfra +10* a2.cyfra +100*a3.cyfra + 1000*a4.cyfra
+ 10000*a5.cyfra + 10000*a6.cyfra AS liczba , a1.cyfra AS cyfra, a1.bit AS bit
FROM Dziesiec a1, Dziesiec a2, Dziesiec a3, Dziesiec a4, Dziesiec a5, Dziesiec a6;
```

4. Testy wydajności

Podczas testów został położony nacisk na efektywność złączeń oraz zapytań zagnieżdżonych na tabelach o względnie sporej objętości. Przedmiotem badań było miedzy innymi: sprawdzenie różnic w wydajności zapytań dotyczących tabel zdenormalizowanych oraz znormalizowanych, różnice w wydajności zapytań podczas gdy tabela nie posiada żadnych indeksów poza tymi będącymi kluczami głównymi a zapytaniami z nałożonymi indeksami na wszystkie wykorzystywane kolumny oraz zamiany w wydajności w różnych systemach bazodanowych .Wykorzystano dwa popularne systemy baz danych: PostgreSQL oraz MySQL. Prace zostały przeprowadzone na komputerze o następującej specyfikacji:

Procesor: Apple M1,
Pamięć ram: 8 GB,
Dysk SSD: 512 GB,
System operacyjny: MacOS big sur wersja 11.4,

Specyfikacja serwerów baz danych:

Wersja PostgreSQL: 2.5.6,
Wersja: MySQL: 8.0.29

Zapytania do bazy danych wykorzystane podczas testów wydajności:

Zapytanie 1: Złączenie syntetycznej tabeli zawierającej milion rekordów z tabelą strygraficzną w postaci zdenormalizowanej, do warunku złączenia dodano operacje modulo, która dopasowuje zakresy wartości złączanych kolumn:

```
SELECT COUNT (*) FROM Milion INNER JOIN GeoTabela
ON (mod(Milion.liczba, 68)=(GeoTabela.id_pietro));
```

Zapytanie 2: Załączenie syntetycznej tabeli miliona wyników z tabelą geochronologiczną w postaci znormalizowanej, reprezentowaną przez złączenia pięciu tabel:

```
SELECT COUNT (*) FROM Milion INNER JOIN GeoPietro  
ON (mod(Milion.liczba,68)=GeoPietro.id_pietro)  
NATURAL JOIN GeoEpoka NATURAL JOIN GeoOkres NATURAL JOIN GeoEra  
NATURAL JOIN GeoEon;
```

Zapytanie 3: Złączenie syntetycznej tablicy miliona wyników z tabelą geochronologiczną w postaci zdenormalizowanej, gdzie złączenie jest wykonywane poprzez zagnieźdżenie skorelowane:

```
SELECT COUNT (*) FROM Milion WHERE mod(Milion.liczba,68)=(SELECT id_pietro  
FROM GeoTabela WHERE mod(Milion.liczba, 68)=(id_pietro));
```

Zapytanie 4: Złączenie syntetycznej tablicy miliona rekordów z tabelą geochronologiczną w postaci znormalizowanej, gdzie złączenie jest wykonywane poprzez zagnieźdżenie skorelowane, a zapytanie wewnętrzne jest złączeniem tabel poszczególnych jednostek geochronologicznych :

```
SELECT COUNT (*) FROM MILION WHERE mod(Milion.liczba, 68) IN  
(SELECT GeoPietro.id_pietro FROM GeoPietro NATURAL JOIN GeoEpoka  
NATURAL JOIN GeoOkres NATURAL JOIN GeoEra NATURAL JOIN GeoEon);
```

5. Wyniki testów wydajności

Nr. Zapytania	PostgreSQL bez nałożonego indeksu					PostgreSQL z nałożonym indeksem				
	Próba 1	Próba 2	Próba 3	Próba 4	Próba 5	Próba 1	Próba 2	Próba 3	Próba 4	Próba 5
1	170	124	123	114	109	138	123	105	111	120
2	181	162	156	148	153	160	152	146	152	148
3	5833	5812	5814	5808	5816	4963	4970	4965	4961	4960
4	114	116	121	127	111	130	110	123	121	119

Rys. 5. Tabela przedstawiająca rezultaty testów w PostgreSQL, jednostka to milisekundy

Nr. Zapytania	MySQL bez nałożonego indeksu					MySQL z nałożony indeksem				
	Próba 1	Próba 2	Próba 3	Próba 4	Próba 5	Próba 1	Próba 2	Próba 3	Próba 4	Próba 5
1	1184	1135	1141	1133	1132	1179	1132	1134	1136	1141
2	13750	13658	13749	13674	13759	11026	11074	11009	11040	11106
3	1509	1506	1497	2498	1497	1504	1507	1564	1506	1505
4	14674	14844	14602	14684	14780	11102	11110	11135	11140	11148

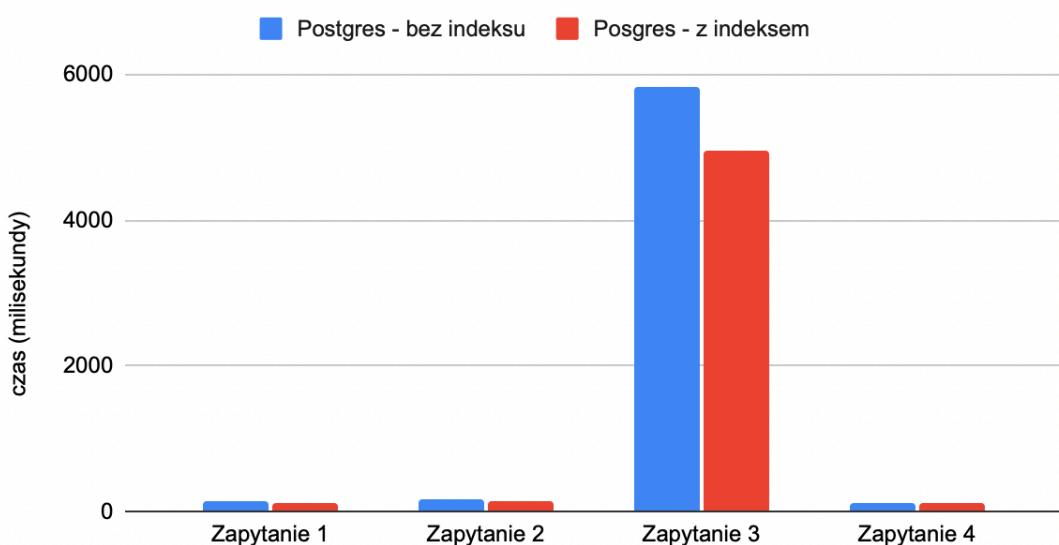
Rys. 6. Tabela zawierająca wyniki testów w MySQL, jednostka to milisekundy

Nr. zapytania	Postgres bez indeksu		Postgres z indeksem		MySQL bez indeksu		MySQL z indeksem	
	Czas min.	Średnia	Czas min.	Średnia	Czas min.	Średnia	Czas min.	Średnia
1	109	128	105	119,4	1132	1145	1132	1144,4
2	148	160	146	151,6	13658	13718	11009	11051
3	5808	5816,6	4960	4963,8	1497	1701,4	1504	1517,2
4	111	117,8	110	120,6	14602	14716,8	11102	11127

Rys. 7. Tabela przedstawiająca podsumowanie wyników z Post oraz MySQL, jednostka to milisekundy

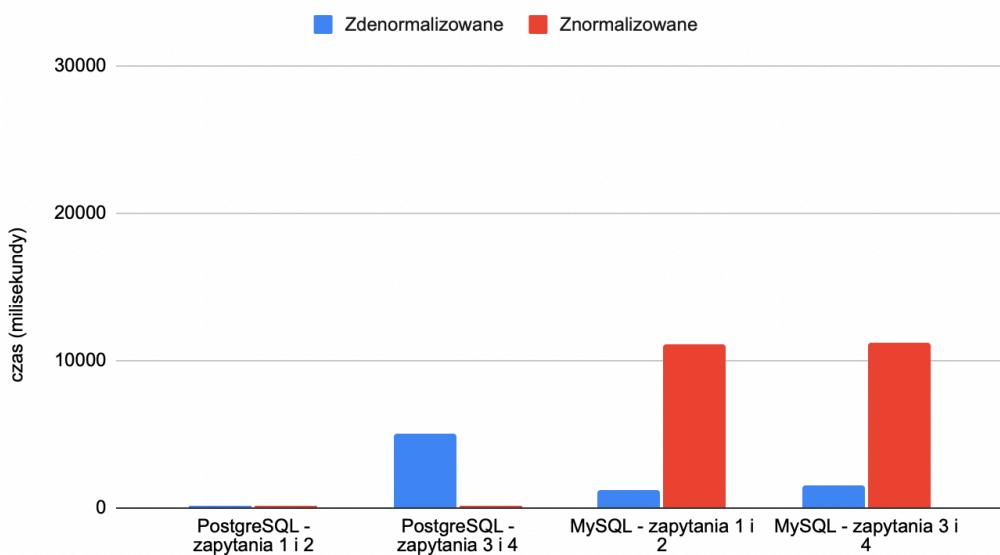
Wizualizacja wybranych danych przedstawionych na rysunkach 5, 6, 7:

Postgres - różnice w czasie wykonania zapytania z nałożonym indeksem oraz bez indeksu



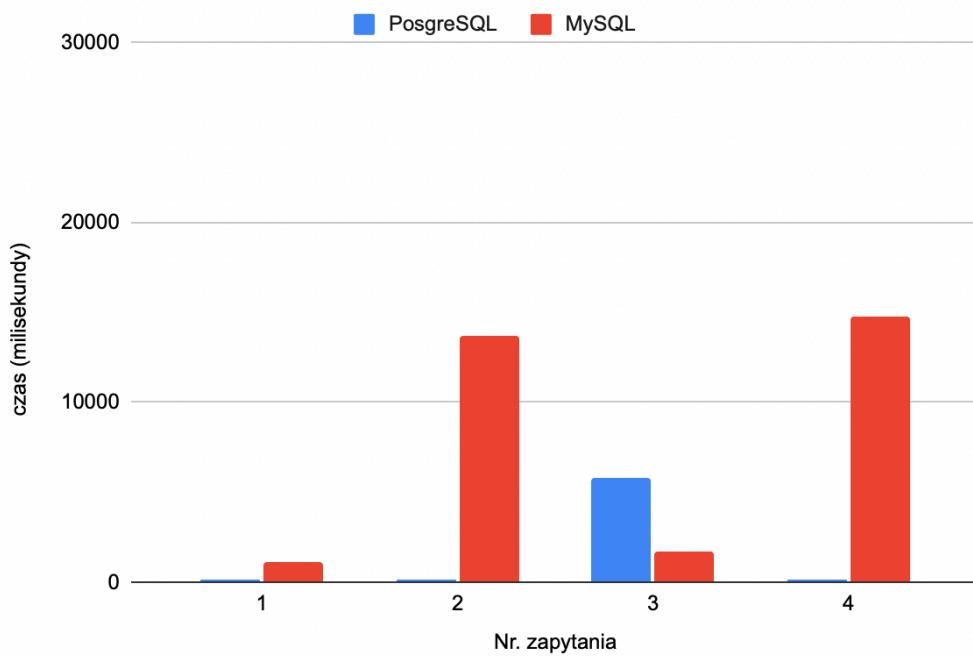
Rys. 8. Różnice w wydajności zapytań dla tabel bez nałożonego indeksu oraz z indeksami w PostgreSQL

Dane zdenormalizowane a znormalizowane, PostgreSQL oraz MySQL



Rys. 9. Zależność czasowa między zapytaniami dotyczącymi tabel zdenormalizowanych oraz w postaci normalnej dla PostgreSQL oraz MySQL, z nałożonym indeksem

Czas wykonania zapytań dla tabel bez indeksów - PostgreSQL oraz MySQL



Rys. 10. Porównanie wydajności PostgreSQL oraz MySQL dla zapytań do tabel bez indeksów

6. Wnioski

- nałożenie indeksu na kolumny wykorzystywane podczas wykonywania zapytań w systemie Postgres w 3 z 4 zapytań dało lepszy, średni rezultat czasowy, aniżeli zapytania przeprowadzone na tabelach, którymi indeksami były jedynie klucze główne tabel. Obserwacje można dokonać na rysunku nr. 8, szczególnie zauważalna równica ma miejsce podczas zapytania nr. 3

- dodanie indeksów do tabel wykorzystanych podczas złączeń w systemie MySQL również dało pozytywny rezultat, podobnie jak w Postgres, w 3 z 4 zapytań udało się podnieść wydajność czyli zmniejszyć czas oczekiwania na odpowiedz na konkretne zapytanie

- zapytania dotyczące tabel znormalizowanych w PostgreSQL okazały się szybsze, natomiast w MySQL przewagę miały zapytania dotyczące tabel zdenormalizowanych

- wykonując zapytania w systemie MySQL lepsze rezultatu czasowe udało się uzyskać wykorzystując do nich tabele w postaci zdenormalizowanej, używanej podczas implantacji schematu gwiazdy. Spadek wydajności dla danych znormalizowanych był znaczący, doskonale widać to na rysunku nr. 9.

- w systemie Postgres miała miejsce nieco inna sytuacja, w przypadku zapytań nr. 1 oraz nr. 2, szybsze było zapytanie dotyczące złączenia z tabelą znormalizowaną, natomiast porównując zapytania nr. 3 i nr. 4, o wiele szybsze okazało się zapytanie nr. 4, gdzie złączenie wykonywane jest przez zagnieżdżenie skorelowane

- wzrost wydajności podczas złączeń z tabelami w postaci znormalizowanej oraz zdenormalizowanej w MySQL względem PostgreSQL był o wiele bardziej przewidywalny oraz liniowy.

- porównanie wydajności zapytań, wykorzystując tabele bez indeksów, przedstawiony na rysunku nr. 10 prezentuje ciekawe rezultaty. Tylko podczas jednego z czterech zapytań, średni czas uzyskany podczas przez Postgres był dłuższy niż w systemie MySQL. Okazuje się, że podczas naszych eksperymentów, system Postgres okazał się wydajniejszy.

7. Podsumowanie

Przeprowadzony eksperiment przyniósł kilka ciekawych wniosków. Z pozoru, nakładanie indeksów na kolumny tabel służy zwiększeniu wydajność całego systemu, przeprowadzony eksperiment dowódł, że nie zawsze tak jest, możemy zaobserwować to na rysunkach 7 oraz 8. Mimo braku zwiększenia wydajności w każdym przypadku, nałożenie indeksów z reguły powinno usprawnić działanie zapytań, szczególnie dotyczy to systemu MySQL, w którym zaobserwowano wzrost wydajności we wszystkich średnich czasach względem średnich czasów zapytań do tabel bez indeksów. Co więcej, przechowywane danych w tabelach w postaci znormalizowanej mimo polepszenia elastyczności schematu bazy danych oraz utrzymaniu go w większej przejrzystości, niekoniecznie jest korzystne dla wydajności. Dane przedstawione na rynku nr. 9 potwierdzają stwierdzenie, że zapytania do tabel w postaci znormalizowanej, charakteryzują się dłuższym czasem wykonania. Dane zebrane podczas przeprowadzania

eksperymentu ukazały również, że system MySQL jest mniej wydajny od PostgreSQL, przyjemniej dla większości danych wykorzystanych w eksperymencie.

8. Bibliografia

- eksperiment został przygotowany w oparciu o sprawozdanie Pana Łukasza Jajeśnica oraz Pana Adama Piórkowskiego

- www.techonthenet.com

- geotyda.pl

- vertabelo.com

Michał Żabiński